

```

%*****%
HRM-II DMA Frame Controller State Machine 19 Aug 05 pak
%*****%

TITLE "HRM-II DMA Frame Controller State Machine";
% On receipt of the 100us tick, sequence thru each slave port and transfer the
required number of packets to the TX fifo using the DMARD machine.

If the entire DMA does not complete before a new RUN issues from the
timing chain, the current DMA is aborted, a DMABGN is issued and a new
cycle starts. %

SUBDESIGN frmctl
(symclk      : INPUT;          % 40 Mhz symbol clock%
reset        : INPUT;          % reset %
zerocnt     : INPUT;          % Cyclic transfer count is zero %
skip         : INPUT;          % Channel skip bit %
run          : INPUT;          % 100us tick from timing %
adv          : INPUT;          % Signal from DMARD to advance to next channel %
dsp          : INPUT;          % DSP access to DMA region (read or write) %
cnl0         : OUTPUT;         % Channel select 0 %
cnl1         : OUTPUT;         % Channel select 1 %
cnl2         : OUTPUT;         % Channel select 2 %
cnl3         : OUTPUT;         % Channel select 3 %
checka       : OUTPUT;         % Channel ready check 0 %
checkb       : OUTPUT;         % Channel ready check 1 %
checkc       : OUTPUT;         % Channel ready check 2 %
checkd       : OUTPUT;         % Channel ready check 3 %
dmabgn      : OUTPUT;         % DMABGN to slave ports %
dodma        : OUTPUT;         % Command DMARD machine to execute DMA cycles %
busy         : OUTPUT;         % DMA Busy %

)
VARIABLE
dmafc: MACHINE OF BITS (p[3..0])
WITH STATES (
    idle = B"0000",
    a = B"0101",
    b = B"0111",
    c = B"0011",
    d = B"0001",
    cka = B"1100",
    ckb = B"0100",
    ckc = B"0110",
    ckd = B"0010",
    delay1 = B"1010",
    delay2 = B"1101",
    delay3 = B"1001",
    delay4 = B"1000",
    void1 = B"1011",
    void2 = B"1110",
    void3 = B"1111"
);
BEGIN
%set up the clock%
dmafc.clk=symclk;
dmafc.reset=reset;

% outputs %
dodma = a # b # c # d;
dmabgn = delay2 # delay3;
busy = !idle;
cnl0 = a;

```

```

cnl1 = b;
cnl2 = c;
cnl3 = d;
checka = cka;
checkb = ckb;
checkc = ckc;
checkd = ckd;

% state transitions %
TABLE
    dmafc,run,skip,zeroCnt,adv,dsp => dmafc;
% PS      RUN SKP ZRO ADV DSP => NS %
    idle,   0,  x,  x,  x,  x => idle;
    idle,   1,  x,  x,  x,  x => delay1;

    delay1, x,  x,  x,  x,  x => delay2; % 25ns pause %
    delay2, x,  x,  x,  x,  x => delay3; % 50ns DMABGN %
    delay3, x,  x,  x,  x,  x => delay4; % 75ns pause %
    delay4, x,  x,  x,  x,  x => cka;

    cka,   x,  x,  x,  x,  1 => cka;      % wait for DSP to clear %
    cka,   x,  0,  x,  x,  0 => a;        % dispatch for channel A %
    cka,   x,  1,  x,  x,  0 => ckb;

    ckb,   x,  x,  x,  x,  1 => ckb;      % wait for DSP to clear %
    ckb,   0,  0,  x,  x,  0 => b;        % normal %
    ckb,   0,  1,  x,  x,  0 => ckc;      % channel programmed for skip %
    ckb,   1,  x,  x,  x,  0 => delay1; % new tick %

    ckc,   x,  x,  x,  x,  1 => ckc;
    ckc,   0,  0,  x,  x,  0 => c;
    ckc,   0,  1,  x,  x,  0 => ckd;
    ckc,   1,  x,  x,  x,  x => delay1;

    ckd,   0,  0,  x,  x,  x => d;
    ckd,   0,  1,  x,  x,  x => idle;
    ckd,   1,  x,  x,  x,  x => delay1;

% PS      RUN SKP ZRO ADV DSP => NS %
    a,    0,  x,  0,  0,  x => a;        % running DMAs %
    a,    0,  x,  0,  1,  x => ckb;      % DMARD says to move on %
    a,    0,  x,  1,  x,  0 => ckb;      % counter is zero. move to next channel. %
    a,    1,  x,  x,  x,  x => delay1; % current channel pre-empted by new tick %

    b,    0,  x,  0,  0,  x => b;
    b,    0,  x,  0,  1,  x => ckc;
    b,    0,  x,  1,  x,  x => ckc;
    b,    1,  x,  x,  x,  x => delay1;

    c,    0,  x,  0,  0,  x => c;
    c,    0,  x,  0,  1,  x => ckd;
    c,    0,  x,  1,  x,  x => ckd;
    c,    1,  x,  x,  x,  x => delay1;

    d,    0,  x,  0,  0,  x => d;        % D DMA and the DSP can never collide %
    d,    0,  x,  0,  1,  x => idle;
    d,    0,  x,  1,  x,  x => idle;
    d,    1,  x,  x,  x,  x => delay1;

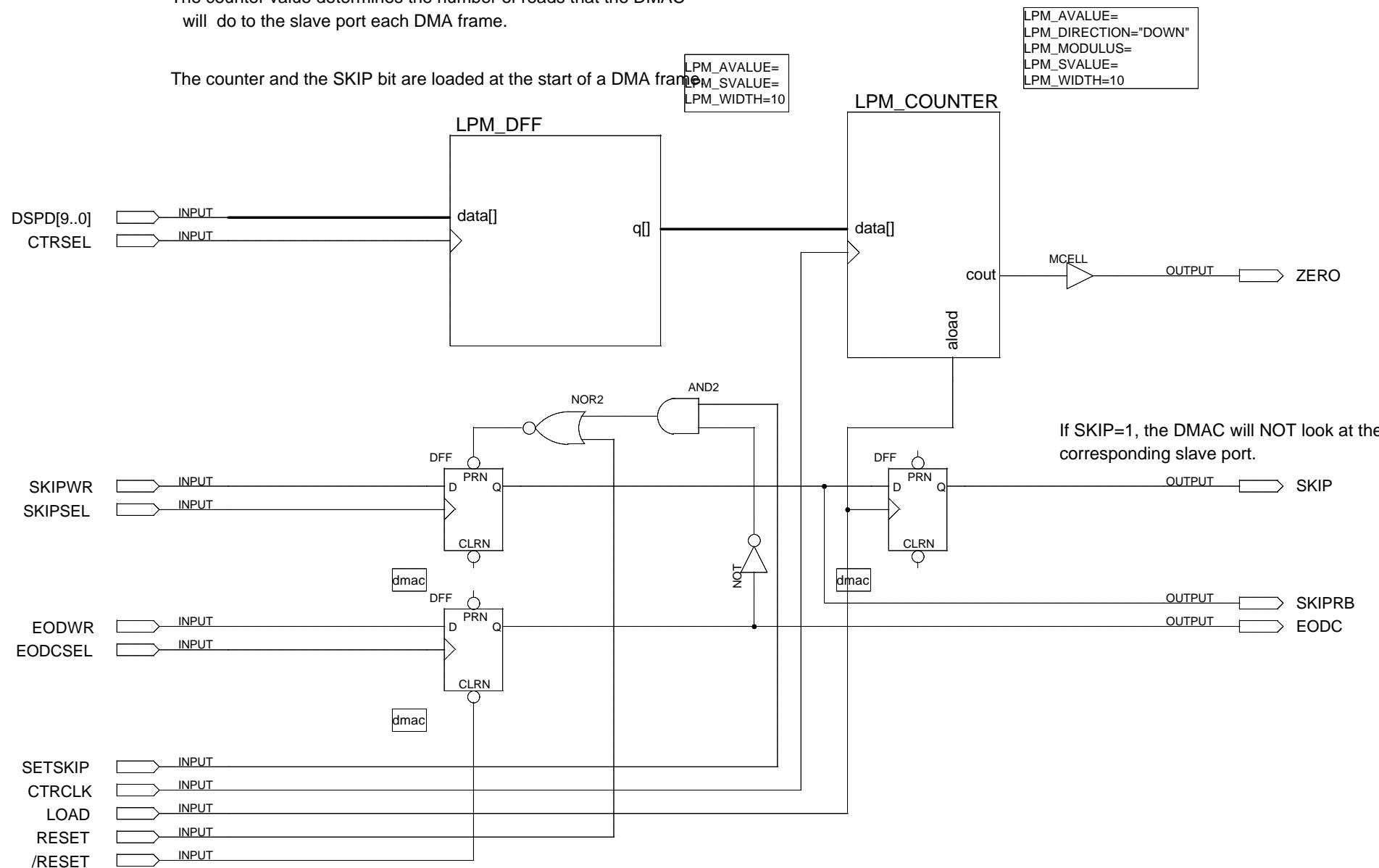
END TABLE;

END;

```

The counter value determines the number of reads that the DMAC will do to the slave port each DMA frame.

The counter and the SKIP bit are loaded at the start of a DMA frame.



If EODControl is clear, /DAV and EOD during a slave read will set the skip bit and the DMAC will ignore the port on subsequent DMA frames.

If EODC is set, the automatic setting of SKIP does not occur.  
RESET sets SKIP and clears EODC.

TITLE	Channel Registers		
COMPANY	Fermilab		
DESIGNER	P. Kasley		
SIZE	C	NUMBER	1.00
DATE	2:52p 10-10-2005	SHEET	1 OF 1

```

%*****%
HRM-II TX FIFO Write State Machine 20 Feb 02 pak
%*****%

TITLE "HRM-II TX FIFO Write State Machine";
% Each time the data latch is clocked, send to the transmit FIFO a header
followed by four symbols. If the FIFO is almost full, hold. %

SUBDESIGN txwrsm
(symclk      : INPUT;           % 40 Mhz symbol clock%
 reset       : INPUT;           % reset %
 dmab_lat   : INPUT;           % DMA bus latch clock %
 txpaf      : INPUT;           % tx fifo almost full (7 spaces remain) %
 sel[2..0]   : OUTPUT;          % tx mux select %
 txwr       : OUTPUT;          % tx FIFO write output %
)
)

VARIABLE
txwrsm: MACHINE OF BITS (p[2..0])
WITH STATES (idle=H"0",
             hdr=H"1",
             data_lo=H"4",
             data_hi=H"5",
             adrs_lo=H"6",
             adrs_hi=H"7",
             wait=H"2",
             unused=H"3");

BEGIN
%set up the clock%
txwrsm.clk=symclk;
txwrsm.reset=reset;

% outputs %
sel[2..0] = p[2..0];
txwr = !idle & !wait;

% state transitions %
TABLE
  txwrsm,dmab_lat,txpaf => txwrsm;

% PS      dmab_lat    txpaf    =>  NS %
idle,    0,          x        =>  idle;
idle,    1,          0        =>  hdr;
idle,    1,          1        =>  wait;
hdr,     x,          x        =>  data_lo;
data_lo,x,          x        =>  data_hi;
data_hi,x,          x        =>  adrs_lo;
adrs_lo,x,          x        =>  adrs_hi;
adrs_hi,0,          x        =>  idle;
adrs_hi,1,          0        =>  hdr;
adrs_hi,1,          1        =>  wait;
wait,    x,          0        =>  hdr;
wait,    x,          1        =>  wait;
unused,  x,          x        =>  idle;
END TABLE;

END;

```

```
%*****  
HRM2 transmit mux 07mar02 pak  
*****%  
  
TITLE "HRM2 Transmit Mux";  
  
SUBDESIGN txmux  
(sel[2..0] : INPUT;          % Mux select %  
 dmad[15..0]: INPUT;        % data word %  
 dmaa[15..0]: INPUT;        % address word %  
 out[8..0]  : OUTPUT;       % output bus %  
)  
  
BEGIN  
  
IF (sel2 == 0) THEN out[7..0]=H"00"      ; END IF;  
IF (sel[] == 4) THEN out[7..0]=dmad[7..0] ; END IF;  
IF (sel[] == 5) THEN out[7..0]=dmad[15..8] ; END IF;  
IF (sel[] == 6) THEN out[7..0]=dmaa[7..0] ; END IF;  
IF (sel[] == 7) THEN out[7..0]=dmaa[15..8] ; END IF;  
  
IF (sel2 == 0) THEN out8=B"1"; ELSE out8=B"0"; END IF;  
END;
```